```
FFFFFFFFFFFFFFF     OOOOOOOOO      RRRRRRRRRRRR     RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
FFFFFFFFFFFFFFF     OOOOOOOOO      RRRRRRRRRRRR     RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
FFFFFFFFFFFFFFF     OOOOOOOOO      RRRRRRRRRRRR     RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
FFF              OOO         OOO   RRR         RRR  RRR         RRR        TTT        LLL
FFF              OOO         OOO   RRR         RRR  RRR         RRR        TTT        LLL
FFF              OOO         OOO   RRR         RRR  RRR         RRR        TTT        LLL
FFF              OOO         OOO   RRR         RRR  RRR         RRR        TTT        LLL
FFF              OOO         OOO   RRR         RRR  RRR         RRR        TTT        LLL
FFF              OOO         OOO   RRR         RRR  RRR         RRR        TTT        LLL
FFFFFFFFFFFFF    OOO         OOO   RRRRRRRRRRRR     RRRRRRRRRRRR           TTT        LLL
FFFFFFFFFFFFF    OOO         OOO   RRRRRRRRRRRR     RRRRRRRRRRRR           TTT        LLL
FFFFFFFFFFFFF    OOO         OOO   RRRRRRRRRRRR     RRRRRRRRRRRR           TTT        LLL
FFF              OOO         OOO   RRR   RRR        RRR   RRR              TTT        LLL
FFF              OOO         OOO   RRR   RRR        RRR   RRR              TTT        LLL
FFF              OOO         OOO   RRR   RRR        RRR   RRR              TTT        LLL
FFF              OOO         OOO   RRR      RRR     RRR      RRR           TTT        LLL
FFF              OOO         OOO   RRR      RRR     RRR      RRR           TTT        LLL
FFF              OOO         OOO   RRR      RRR     RRR      RRR           TTT        LLL
FFF                 OOOOOOOOO      RRR         RRR  RRR         RRR        TTT        LLLLLLLLLLLLLLL
FFF                 OOOOOOOOO      RRR         RRR  RRR         RRR        TTT        LLLLLLLLLLLLLLL
FFF                 OOOOOOOOO      RRR         RRR  RRR         RRR        TTT        LLLLLLLLLLLLLLL
```

```
**FILE**ID**FORCB                                    E  7



FFFFFFFFFF    000000    RRRRRRRR      CCCCCCCC  BBBBBBBB
FFFFFFFFFF    000000    RRRRRRRR      CCCCCCCC  BBBBBBBB
FF          00      00  RR      RR  CC          BB      BB
FF          00      00  RR      RR  CC          BB      BB
FF          00      00  RR      RR  CC          BB      BB
FFFFFFFF    00      00  RRRRRRRR    CC          BBBBBBBB
FFFFFFFF    00      00  RRRRRRRR    CC          BBBBBBBB
FF          00      00  RR  RR      CC          BB      BB
FF          00      00  RR  RR      CC          BB      BB
FF          00      00  RR    RR    CC          BB      BB      ....
FF          00      00  RR    RR    CC          BB      BB      ....
FF            000000    RR      RR    CCCCCCCC  BBBBBBBB        ....
FF            000000    RR      RR    CCCCCCCC  BBBBBBBB        ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

```
0 MODULE FOR$$CB (%TITLE 'Push, Pop, Allocate, and deallocate LUB/ISB/RAB'
0                 IDENT = '2-005'          ! File: FORCB.B32  Edit: LEB2005
0                 ) =
1 BEGIN
1 !
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
1 !*  ALL RIGHTS RESERVED.                                                    *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
1 !*  TRANSFERRED.                                                            *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
1 !*  CORPORATION.                                                            *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1
1 !++
1 ! FACILITY: language support library
1 !
1 ! ABSTRACT:
1 !
1 !       This module interfaces to FOR$$CCB_DATA to allocate,
1 !       deallocate, push and pop the LUB/ISB/RAB data structure, which
1 !       is central to the I/O system.
1 !
1 ! ENVIRONMENT: User mode, AST level or not or mixed
1 !
1 ! AUTHOR:  Thomas N. Hastings, CREATION DATE: 01-June-77
1 !
1 ! MODIFIED BY:
1 !
1 !       Thomas N. Hastings, 01-June-77: VERSION 01
1 ! [Previous edit history removed.  SBL 24-Sept-1982]
1 ! 1-032 - Remove AST reentrancy window by performing IOINPROG interlock before
1 !         LUN_OWNR test in FOR$$CB_PUSH.  Replace individual zeroing of ISB
1 !         bits with a zero of the word in which they are contained for better
1 !         code.  Use a new structure for OTS$$V_LUN_OWNR for smaller code.
1 !         SBL 25-Sept-1980
1 ! 1-033 - Include secondary message FOR$_IO_NONFOR when signaling
1 !         FOR$K_MIXFILACC.  JAW 29-Aug-1981
1 ! 1-034 - Clear OTS$$V_IOINPROG before signaling FOR$K_MIXFILACC, to
1 !         ensure that unit is left in a consistent state.  SPR 11-38566.
1 !         JAW 29-Aug-1981
1 ! 1-035 - Replace $DESCRIPTOR in edit 1-033 with UPLIT to keep code PIC.
1 !         JAW 31-Aug-1981
```

```
:    58         0058  1 ! 1-036 - Add missing external declarations.  SBL 2-Dec-1981
:    59         0059  1 ! 2-001 - Remove all references to OTS$$ routines and data structures.
:    60         0060  1 !         The data structures are now FORTRAN-only, although their layout
:    61         0061  1 !         and use is still in parallel with BASIC's.  Change to use
:    62         0062  1 !         prologue file, and general cleanup for inclusion in FORRTL
:    63         0063  1 !         shared image.  SBL 24-Sept-1982
:    64         0064  1 ! 2-002 - Use ISB$A_PREVIOUS_LUB for backlink to previous LUB instead of
:    65         0065  1 !         second entry in FOR$$AA_LUB_TAB.  Add logic to allow simultaneous
:    66         0066  1 !         ENCODE/DECODE/Internal File operations.  SBL 2-Dec-1982
:    67         0067  1 ! 2-003 - Allocate FAB and NAM along with RAB and rest of CCB from heap.
:    68         0068  1 !         SBL 17-Jan-1983
:    69         0069  1 ! 2-004 - Deallocate RFA cache if necessary.  SBL 2-June-1983
:    70         0070  1 ! 2-005 - Change reference in DEALLOCATE from LUB$A_RFA_CACHE_PTR to
:    71         0071  1 !         LUB$A_RFA_CACHE_BEG to fix problem with BACKSPACE.
:    72         0072  1 !         LEB 27-Jan-1984
:    73         0073  1 !--
:    74         0074  1
```

```
%SBTTL'Declarations'
!
! PROLOGUE FILE:
!

REQUIRE 'RTLIN:FORPROLOG';                              ! Structure and symbol definitions
!
! TABLE OF CONTENTS:
!

FORWARD ROUTINE
    FOR$$CB_PUSH : JSB_CB_PUSH NOVALUE,                 ! Allocate or find  LUB/ISB/RAB - beg of each I/O statment
    ALLOCATE : CALL_CCB NOVALUE,                        ! Allocate CCB
    FOR$$CB_POP : JSB_CB_POP NOVALUE,                   ! Pop LUB/ISB/RAB - end of each I/O statement
    DEALLOCATE : CALL_CCB NOVALUE,                      ! Deallcoate CCB
    FOR$$CB_GET : JSB_CB_GET NOVALUE,                   ! Get current LUB/ISB/RAB (called by non-shared code only)
    FOR$$CB_FETCH : CALL_CCB NOVALUE,                   ! Fetch a LUB, or 0
    FOR$$NEXT_LUN : NOVALUE,                            ! Get next FORTRAN LUN.
    FOR$$FP_MATCH : CALL_CCB NOVALUE,                   ! Get CCB that matches FP
    INITIALIZE_INTFIL_QUEUE: NOVALUE;                   ! Initialize INTFIL_QUEUE

!+
! Include FOR$$CB_RET as a synonym for FOR$$CB_POP to maintain
! compatability with old versions of FOR$$ERROR.
!-

GLOBAL BIND
    ROUTINE
    FOR$$CB_RET = FOR$$CB_POP : JSB_CB_POP NOVALUE;

!
! GLOBAL STORAGE:
!

GLOBAL
    FOR$$A_CUR_LUB : INITIAL (0);       ! Contains the address of the current LUB

!+
! The following structure is used for addressing FOR$$AA_LUB_TAB.
! It is similar to VECTOR, but offsets the index so that
! negative logical unit numbers can be used.
!-

STRUCTURE
    FOR$$LUB_TAB_ST [I; N, LB, UNIT = 4, EXT = 0] =
        [N*UNIT]
        (FOR$$LUB_TAB_ST + ((I - LB)*UNIT))<0, %BPUNIT*UNIT, EXT>;

!+
! The following table of longwords is used to associate LUB addresses with
! unit numbers.  Each entry contains 0 if there is no
! LUB, or the address of the LUB.
!-

GLOBAL
    FOR$$AA_LUB_TAB : VOLATILE FOR$$LUB_TAB_ST
```

```
1           [-LUB$K_ILUN_MIN + LUB$K_LUN_MAX + 1, LUB$K_ILUN_MIN];
1
1  !
1  ! OWN STORAGE:
1  !
1
1  !+
1  ! Each bit of the following BITVECTOR corresponds to a LUN.  The bit is
1  ! set if there is any I/O activity outstanding for the LUN.  The bit
1  ! must be kept here rather than in the LUB because there can be I/O
1  ! activity outstanding even before the LUB is allocated.
1  !
1  ! The name FOR$$V_IOINPROG is bound to the appropriate offset in the
1  ! bitvector so that the correct bit can be directly addressed by unit number.
1  !-
1
1  OWN
1      IOINPROG_VECTOR : VOLATILE BITVECTOR
1          [((-[UB$K_ILUN_MIN + LUB$K_LUN_MAX + %BPVAL)/%BPVAL)*%BPVAL];
1  BIND
1      FOR$$V_IOINPROG = IOINPROG_VECTOR [((7-LUB$K_ILUN_MIN)/8)*8]:
1          VO[ATILE BITVECTOR [];
1
1  !+
1  ! The following is a queue (non-interlocked) that holds LUBs for ENCODE/DECODE
1  ! and internal file operations.  This permits more than one of these operations
1  ! to be active simultaneously.
1  !-
1
1  OWN
1      INTFIL_QUEUE: VOLATILE VECTOR [2] INITIAL (0,0),
1      V_INTFIL_QUEUE_INIT: VOLATILE INITIAL (0);  ! 1 when queue initialized
1
1  !
1  ! EXTERNAL REFERENCES:
1  !
1
1  EXTERNAL ROUTINE
1      FOR$$ERRSNS_SAV : NOVALUE,
1      FOR$$SIG_NO_LUB : NOVALUE,                       ! convert FORTRAN err # to 32-bit code
1                                                       ! Pass LUN explicitly since no current LUB.
1                                                       ! and call LIB$STOP. should never return
1      FOR$$SIG_DATCOR : NOVALUE,                       ! SIGNAL_STOP OTS$_INTDATCOR (INTERNAL
1                                                       ! DATA CORRUPTED IN RUN-TIME LIBRARY)
1                                                       ! in FORTRAN environment
1      FOR$$SIGNAL_STO : NOVALUE,                       ! Signal a fatal FORTRAN error
1      FOR$$GET_VM,                                     ! Get virtual memory
1      FOR$$FREE_VM : NOVALUE;                          ! Free virtual memory
1
```

```
1 GLOBAL ROUTINE FOR$$CB_PUSH (%SBTTL'Allocate or find CCB'
1          LOGICAL_UNIT,                          ! Logical unit no. (by-value)
1          LUN_MIN)                               ! Minimum logical unit number (by-value)
1    : JSB_CB_PUSH NOVALUE =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !        FOR$$CB_PUSH checks for legal logical UNIT number
1 !        which varyies depending on whether this is OPEN or
1 !        default open.  If logical_unit already has
1 !        a LUB/ISB/RAB allocated, only part of the per I/O statement part
1 !        of LUB/ISB/RAB is cleared, namely just the status bits in ISB.
1 !        Otherwise virtual memory is allocated for this logical_unit
1 !        and the entire block is initialized to 0.  Then the allocated address
1 !        is remembered in OWN table FOR$$A_LUB_TAB indexed by
1 !        logical_unit.  The RAB is initialized to constants which
1 !        do not change during execution.
1 !
1 !        If an I/O statement on this unit is already in progress, this
1 !        routine signals an error and does not return.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !        JSB FOR$$CB_PUSH (R2=logical_unit.rl.v, R0=lun_min.rl.v)
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !        LOGICAL_UNIT.rl.v       Value of logical unit for which LUB/ISB/RAB is desired (signed)
1 !                                May be negative for TYPE, ACCEPT, READ, PRINT
1 !        LUN_MIN.rl.v            Value of minimum legal logical unit number (signed)
1 !                                Since in a register, must be present.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !        FOR$$AA_LUB_TAB[logical_unit]   Adr. of LUB/ISB/RAB or 0 for
1 !                                        this unit
1 !        FOR$$V_IOINPROG[logical unit]   I/O in progress flag
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !        CCB                             Base pointer set to adr. of LUB/ISB/RAB for logical_unit.
1 !        FOR$$AA_LUB_TAB[logical_unit]   Adr. of LUB/ISB/RAB for logical_unit
1 !        LUB$W_LUN                       signed logical unit number
1 !        RAB$B_BID
1 !        RAB$B_BLN
1 !        RAB$V_TPT                       1
1 !        RAB$V_RAH                       1
1 !        RAB$V_WBH                       1
1 !        RAB$V_LOC                       1
1 !
1 ! ROUTINE VALUE:
1 !
1 !        None
1 !
1 ! SIDE EFFECTS:
1 !
```

```
!	    Allocates virtual memory if needed.
!	    SIGNAL_STOPs FOR$_RECIO_OPE (40='RECURSIVE I/O OPERATION') if
!	    logical_unit already is in the middle of an I/O statement
!	    SIGNAL_STOPs FOR$_INVLOGUNI (32='INVALID LOGICAL UNIT NUMBER')
!	    if logical_unit is out of range.
!	    SIGNAL_STOPs FOR$_INSVIRMEM (41='INSUFFICIENT VIRTUAL MEMORY')
!	    if cannot expand program region if needed.
!--

    BEGIN

    BUILTIN
	TESTBITSS;

    EXTERNAL REGISTER
	CCB : REF $FOR$CCB_DECL;

    !+
    ! Check range of logical_unit.  If out of range,
    ! SIGNAL_STOP FOR$_INVLOGUNI (32='INVALID LOGICAL UNIT NUMBER')
    !-

    IF ((.LOGICAL_UNIT GTR LUB$K_LUN_MAX) OR (.LOGICAL_UNIT LSS .LUN_MIN))
    THEN
	BEGIN
	FOR$$SIG_NO_LUB (FOR$K_INVLOGUNI, .LOGICAL_UNIT);
	RETURN;
	END;

    !+
    ! Test and set IO in progress interlock before doing anything else!
    ! If this is ENCODE/DECODE/Internal File, ignore interlock.
    !-

    IF (TESTBITSS (FOR$$V_IOINPROG [.LOGICAL_UNIT]))
    THEN
	IF .LOGICAL_UNIT NEQ LUB$K_LUN_ENCD
	THEN
	    BEGIN
	    FOR$$SIG_NO_LUB (FOR$K_RECIO_OPE, .LOGICAL_UNIT);
	    RETURN;
	    END;

    !+
    ! The following assignment generates no code, but it causes BLISS to generate
    ! optimal code for the remainder of the routine by preventing the CSE
    ! .LOGICAL_UNIT-LUB$K_ILUN_MIN from being bound to R2.  Thanks, and a tip
    ! of the keyboard to Steve Hobbs.
    !-

    LOGICAL_UNIT = .LOGICAL_UNIT;

    !+
    ! Get the CCB address for this unit.
    !-

    CCB = .FOR$$AA_LUB_TAB [.LOGICAL_UNIT];
```

```
2
2      !+
2      ! Allocate a LUB/ISB/RAB if necessary.
2      !-
2
2      IF .CCB EQLA 0
2      THEN
2          ALLOCATE (.LOGICAL_UNIT)
2      ELSE
2      !+
2      ! LUB/ISB/RAB already allocated.  Perform sanity check.
2      !-
3          BEGIN
3
4          IF ((.CCB [LUB$W_LUN] NEQU .LOGICAL_UNIT<0,16,1>) OR
4              (.CCB [RAB$B_BID] NEQU RAB$C_BID))
3          THEN
3              FOR$$SIG_DATCOR ();
2          END;
2
2      !+
2      ! Initialize certain ISB fields, to save FOR$$IO_BEG the trouble.
2      !-
2
2      CCB [ISB$W_STTM_STAT] = 0;
2      CCB [ISB$W_FMT_LEN] = 0;
2      CCB [ISB$A_USER_FP] = 0;
2
2      !+
2      ! Link in previous LUB and make this LUB the current one.
2      !-
2
2      CCB [ISB$A_PREVIOUS_LUB] = .FOR$$A_CUR_LUB;
2      FOR$$A_CUR_LUB = .CCB;
2
2      !+
2      ! Return with register CCB loaded.
2      !-
2
2      RETURN;
1      END;                                        ! End of routine FOR$$CB_PUSH

;                                                     .TITLE  FOR$$CB Push, Pop, Allocate, and deallocate LUB
                                                                      /ISB/RAB
                                                      .IDENT  \2-005\

                                                      .PSECT  _FOR$DATA,NOEXE,  PIC,2

                             00000000  00000 FOR$$A_CUR_LUB::
                                                      .LONG   0                                          ;
                                       00004 FOR$$AA_LUB_TAB::
                                                      .BLKB   512
                                       00204 IOINPROG_VECTOR:
                                                      .BLKB   16
                   00000000  00000000  00214 INTFIL_QUEUE:
                                                      .LONG   0, 0                                       ;
```

```
                                           00000000  0021C V_INTFIL_QUEUE_INIT:
                                                                   .LONG    0                                                  ;

                                                           FOR$$V_IOINPROG=      IOINPROG_VECTOR+1
                                                                   .EXTRN   FOR$$ERRSNS_SAV
                                                                   .EXTRN   FOR$$SIG_NO_LUB
                                                                   .EXTRN   FOR$$SIG_DATCOR
                                                                   .EXTRN   FOR$$SIGNAL_STO
                                                                   .EXTRN   FOR$$GET_VM, FOR$$FREE_VM

                                                                   .PSECT   _FOR$CODE,NOWRT,  SHR,  PIC,2

                        00000077   8F              52   D1 00000 FOR$$CB_PUSH::
                                                                   CMPL     LOGICAL_UNIT, #119                               ; 0325
                                                   05   14 00007           BGTR     1$                                                   ;
                                   50              52   D1 00009           CMPL     LOGICAL_UNIT, LUN_MIN                            ;
                                                   06   18 0000C           BGEQ     2$                                                   ;
                                                   52   DD 0000E 1$:       PUSHL    LOGICAL_UNIT                                     ; 0328
                                                   20   DD 00010           PUSHL    #32                                                  ;
                                                   15   11 00012           BRB      3$                                                   ;
                     15 00000000'     EF           52   E3 00014 2$:       BBCS     LOGICAL_UNIT, FOR$$V_IOINPROG, 4$                ; 0337
                        FFFFFFFB   8F              52   D1 0001C           CMPL     LOGICAL_UNIT, #-5                                ; 0339
                                                   0C   13 00023           BEQL     4$                                                   ;
                                                   52   DD 00025           PUSHL    LOGICAL_UNIT                                     ; 0342
                                                   28   DD 00027           PUSHL    #40                                                  ;
                        00000000G  00              02   FB 00029 3$:       CALLS    #2, FOR$$SIG_NO_LUB                              ;
                                                   05   05 00030           RSB                                                           ; 0341
                                   5B 00000000'EF42     D0 00031 4$:       MOVL     FOR$$AA_LUB_TAB+32[LOGICAL_UNIT], CCB            ; 0359
                                                   09   12 00039           BNEQ     5$                                                   ; 0365
                                                   52   DD 0003B           PUSHL    LOGICAL_UNIT                                     ; 0367
                            0000V  CF              01   FB 0003D           CALLS    #1, ALLOCATE                                     ;
                                                   12   11 00042           BRB      7$                                                   ;
                                   52        C6    AB   B1 00044 5$:       CMPW     -58(CCB), LOGICAL_UNIT                           ; 0374
                                                   05   12 00048           BNEQ     6$                                                   ;
                                   01              6B   91 0004A           CMPB     (CCB), #1                                        ; 0375
                                                   07   13 0004D           BEQL     7$                                                   ;
                        00000000G  00              00   FB 0004F 6$:       CALLS    #0, FOR$$SIG_DATCOR                              ; 0377
                                             96    AB   B4 00056 7$:       CLRW     -106(CCB)                                        ; 0384
                                           FF72    CB   B4 00059           CLRW     -142(CCB)                                        ; 0385
                                           FF4C    CB   D4 0005D           CLRL     -180(CCB)                                        ; 0386
                             FF48  CB 00000000'    EF   D0 00061           MOVL     FOR$$A_CUR_LUB, -184(CCB)                        ; 0392
                        00000000'  EF              5B   D0 0006A           MOVL     CCB, FOR$$A_CUR_LUB                              ; 0393
                                                   05   05 00071           RSB                                                           ; 0400

; Routine Size:  114 bytes,    Routine Base:  _FOR$CODE + 0000


;  338          0401  1
```

```
1 ROUTINE ALLOCATE (%SBTTL'Allocate CCB'
1         LOGICAL_UNIT                              ! LUN to which to allocate the CCB
1     ) : CALL_CCB NOVALUE =                        ! Allocate LUB/ISB/RAB
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       Allocate heap storage for the LUB/ISB/RAB/FAB/NAM.  This is done
1 !       the first time a logical unit is referenced, and the first
1 !       time after a CLOSE.
1 !
1 !       If this is an ENCODE/DECODE/Internal File, try getting a "short LUB"
1 !       from Q_INTFIL_QUEUE.  If empty, allocate a short LUB.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       ALLOCATE (.LOGICAL_UNIT)
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       LOGICAL_UNIT.rl.v       LUN to which to allocate the CCB
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       INTFIL_QUEUE            Queue of internal file LUBs
1 !
1 ! IMPLICIT OUTPUTS
1 !
1 !       FOR$$AA_LUB_TAB [.LOGICAL_UNIT] and CCB are set
1 !
1 ! SIDE EFFECTS:
1 !
1 !       Allocates virtual storage.
1 !       Signals if virtual storage is exhausted.
1 !
1 !--
1
2    BEGIN
2
2    EXTERNAL REGISTER
2        CCB : REF $FOR$CCB_DECL;
2
2    BIND
2        FAB = CCB: REF $FOR$FAB_CCB_STRUCT,
2        NAM = CCB: REF $FOR$NAM_CCB_STRUCT;
2
2    BUILTIN
2        REMQUE;
2
2    !+
2    ! Split depending on whether or not this is an internal file.
2    !-
2
2    IF .LOGICAL_UNIT NEQ LUB$K_LUN_ENCD
2    THEN
3        BEGIN
3
```

```
3               !+
3               ! This is not an internal file or ENCODE/DECODE.  Allocate a full-length
3               ! LUB from heap storage and initialize it.
3               !-
3
4               CCB = FOR$$GET_VM ((ISB$K_ISB_LEN + LUB$K_LUB_LEN + RAB$C_BLN +
3                   FAB$C_BLN + NAM$C_BLN), .LOGICAL_UNIT);
3               CH$FILL (0, LUB$K_LUB_LEN + RAB$C_BLN + FAB$C_BLN + NAM$C_BLN,
3                   .CCB + ISB$K_ISB_LEN);
3               CCB = .CCB + ISB$K_ISB_LEN + LUB$K_LUB_LEN;
3               CCB [LUB$W_LUN] = .LOGICAL_UNIT;
3               CCB [RAB$B_BID] = RAB$C_BID;
3               CCB [RAB$B_BLN] = RAB$C_BLN;
3               FAB [FAB$B_BID] = FAB$C_BID;
3               FAB [FAB$B_BLN] = FAB$C_BLN;
3               NAM [NAM$B_BID] = NAM$C_BID;
3               NAM [NAM$B_BLN] = NAM$C_BLN;
3               CCB [RAB$L_FAB] = FAB [0,0,0,0];
3
3               CCB [RAB$V_TPT] = 1;
3               CCB [RAB$V_RAH] = 1;
3               CCB [RAB$V_WBH] = 1;
3               CCB [RAB$V_LOC] = 1;
3               FOR$$AA_LUB_TAB [.LOGICAL_UNIT] = .CCB;
3               RETURN;
3               END;
2
2           !+
2           ! This is an internal file or ENCODE/DECODE.  First check to see if the
2           ! queue of LUBs has been intialized.  If not, initialize it.
2           !-
2
2           IF NOT .V_INTFIL_QUEUE_INIT
2           THEN
2               INITIALIZE_INTFIL_QUEUE ();
2
2           !+
2           ! Try to remove a LUB from the head of the queue.  If empty,
2           ! allocate one instead.
2           !-
2
2           IF REMQUE (.INTFIL_QUEUE [0], CCB)
2           THEN
3               BEGIN
3               !+
3               ! Queue was empty.  Allocate a short LUB and initialize it.
3               !-
3
3               CCB = FOR$$GET_VM ((ISB$K_ISB_LEN + LUB$K_LUB_LEN + RAB$C_BLN),
3                   .LOGICAL_UNIT);
3               CH$FILL (0, LUB$K_LUB_LEN + RAB$C_BLN, .CCB + ISB$K_ISB_LEN);
3               CCB = .CCB + ISB$K_ISB_LEN + LUB$K_LUB_LEN;
3               CCB [LUB$W_LUN] = .LOGICAL_UNIT;
3               CCB [RAB$B_BID] = RAB$C_BID;
3               CCB [LUB$V_DEALLOC] = 1;          ! Force "deallocation" on POP
3               END
2           ELSE
```

```
;  454          0516  2         CCB = .CCB + ISB$K_ISB_LEN + LUB$K_LUB_LEN;     ! Get right base for CCB
;  455          0517  2
;  456          0518  2     RETURN;     ! With LUB address in CCB
;  457          0519  1     END;



                                              00FC 00000 ALLOCATE:
                                                                     .WORD   Save R2,R3,R4,R5,R6,R7                            ; 0402
                            57 00000000G   00   9E 00002             MOVAB   FOR$$GET_VM, R7                                   ;
                            56 00000000'   EF   9E 00009             MOVAB   FOR$$AA_LUB_TAB+32, R6                            ;
              FFFFFFFB      8F        04   AC   D1 00010             CMPL    LOGICAL_UNIT, #-5                                 ; 0455
                                           4A   13 00018             BEQL    1$                                                ;
                                      04   AC   DD 0001A             PUSHL   LOGICAL_UNIT                                      ; 0465
                            7E      0214   8F   3C 0001D             MOVZWL  #532, -(SP)                                       ; 0464
                            67             02   FB 00022             CALLS   #2, FOR$$GET_VM                                   ;
                            5B             50   D0 00025             MOVL    R0, CCB                                           ;
  0158   8F          00     6E             00   2C 00028             MOVC5   #0, (SP), #0, #344, 188(CCB)                      ; 0467
                                    00BC   CB      0002F                                                                       ;
                            5B      0120   CB   9E 00032             MOVAB   288(R11), CCB                                     ; 0468
                       C6   AB        04   AC   B0 00037             MOVW    LOGICAL_UNIT, -58(CCB)                            ; 0469
                            6B      4401   8F   B0 0003C             MOVW    #17409, (CCB)                                     ; 0470
                       44   AB      5003   8F   B0 00041             MOVW    #20483, 68(CCB)                                   ; 0472
                     0094   CB      6002   8F   B0 00047             MOVW    #24578, 148(CCB)                                  ; 0474
                       3C   AB        44   AB   9E 0004E             MOVAB   68(CCB), 60(CCB)                                  ; 0476
                       04   AB 00010602    8F   C8 00053             BISL2   #67074, 4(CCB)                                    ; 0481
                            50        04   AC   D0 0005B             MOVL    LOGICAL_UNIT, R0                                  ; 0482
                          6640             5B   D0 0005F             MOVL    CCB, FOR$$AA_LUB_TAB+32[R0]                       ;
                                                04 00063             RET                                                       ; 0457
                            05      01F8   C6   E8 00064 1$:         BLBS    V_INTFIL_QUEUE_INIT, 2$                           ; 0491
                    0000V   CF             00   FB 00069             CALLS   #0, INITIALIZE_INTFIL_QUEUE                       ; 0493
                            5B      01F0   D6   0F 0006E 2$:         REMQUE  @INTFIL_QUEUE, CCB                                ; 0500
                                           2A   1C 00073             BVC     3$                                                ;
                                      04   AC   DD 00075             PUSHL   LOGICAL_UNIT                                      ; 0508
                            7E      0164   8F   3C 00078             MOVZWL  #356, -(SP)                                       ; 0507
                            67             02   FB 0007D             CALLS   #2, FOR$$GET_VM                                   ;
                            5B             50   D0 00080             MOVL    R0, CCB                                           ;
  00A8   8F          00     6E             00   2C 00083             MOVC5   #0, (SP), #0, #168, 188(CCB)                      ; 0509
                                    00BC   CB      0008A                                                                       ;
                            5B      0120   CB   9E 0008D             MOVAB   288(R11), CCB                                     ; 0510
                       C6   AB        04   AC   B0 00092             MOVW    LOGICAL_UNIT, -58(CCB)                            ; 0511
                            6B             01   90 00097             MOVB    #1, (CCB)                                         ; 0512
                       FF   AB             10   88 0009A             BISB2   #16, -1(CCB)                                      ; 0513
                                                04 0009E             RET                                                       ; 0500
                            5B      0120   CB   9E 0009F 3$:         MOVAB   288(R11), CCB                                     ; 0516
                                                04 000A4             RET                                                       ; 0519

; Routine Size:  165 bytes,    Routine Base:  _FOR$CODE + 0072


;  458          0520  1
```

```
GLOBAL ROUTINE FOR$$CB_POP      %SBTTL'Pop current CCB'
    : JSB_CB_POP NOVALUE =

!++
! FUNCTIONAL DESCRIPTION:
!
!       FOR$$CB_POP pops the curents LUB/ISB/RAB and restores the
!       previous pushed down LUB/ISB/RAB, if any (usually none).
!       Flags old current LUB/ISB/RAB as no longer having as active I/O statement
!
! CALLING SEQUENCE:
!
!       JSB FOR$$CB_POP ()
!
! FORMAL PARAMETERS:
!
!       NONE
!
! IMPLICIT INPUTS:
!
!       CCB                     Adr. of current LUB/ISB/RAB
!
! IMPLICIT OUTPUTS:
!
!       CCB                     Set to 0 (to catch attempt to reference after a pop).
!
! RETURN VALUE:
!
!       NONE
!
! SIDE EFFECTS:
!
!       Changes entire I/O system to another logical unit or none at all
!       SIGNAL_STOPs FORTRAN INTERNAL ERROR if CB was not active.
!--

    BEGIN

    BUILTIN
        TESTBITCC;

    EXTERNAL REGISTER
        CCB : REF $FOR$CCB_DECL;

    LOCAL
        LOGICAL_UNIT;

    !+
    ! Pop this CCB.
    !-

    LOGICAL_UNIT = .CCB [LUB$W_LUN];
    FOR$$A_CUR_LUB = .CCB [ISB$A_PREVIOUS_LUB];

    !+
    ! Deallocate run-time format
    !-
```

```
;   517          0578  2
;   518          0579  3          IF (.CCB [ISB$W_FMT_LEN] NEQ 0)
;   519          0580  2          THEN
;   520          0581  3              BEGIN
;   521          0582  3              FOR$$FREE_VM (.CCB [ISB$W_FMT_LEN], .CCB [ISB$A_FMT_BEG]);
;   522          0583  3              CCB [ISB$W_FMT_LEN] = 0;
;   523          0584  3              CCB [ISB$A_FMT_BEG] = 0;
;   524          0585  2              END;
;   525          0586  2
;   526          0587  2          !+
;   527          0588  2          ! Deallocate this LUB if requested to.
;   528          0589  2          !-
;   529          0590  2
;   530          0591  3          IF (.CCB [LUB$V_DEALLOC])
;   531          0592  2          THEN
;   532          0593  2              DEALLOCATE (.LOGICAL_UNIT);
;   533          0594  2
;   534          0595  2          !+
;   535          0596  2          ! Flag old current LUB/ISB/RAB as no longer having
;   536          0597  2          ! an I/O statement in progress.
;   537          0598  2          ! If LUB was not active, then signal OTS$_INTDATCOR (INTERNAL DATA
;   538          0599  2          ! CORRUPTED IN RUN-TIME LIBRARY).
;   539          0600  2          !-
;   540          0601  2
;   541          0602  3          IF (TESTBITCC (FOR$$V_IOINPROG [.LOGICAL_UNIT]))
;   542          0603  2          THEN
;   543          0604  2              IF .LOGICAL_UNIT NEQU LUB$K_LUN_ENCD
;   544          0605  2              THEN
;   545          0606  2                  FOR$$SIG_DATCOR ();
;   546          0607  2
;   547          0608  2          CCB = 0;
;   548          0609  2
;   549          0610  2          RETURN;
;   550          0611  2
;   551          0612  1          END;                                          ! End of FOR$$CB_POP routine




                          7E        C6   AB  32 00000 FOR$$CB_POP::
                                                          CVTWL   -58(CCB), LOGICAL_UNIT                               ; 0572
                  00000000'  EF     FF48   CB  D0 00004            MOVL    -184(CCB), FOR$$A_CUR_LUB                            ; 0573
                             50     FF72   CB  3C 0000D            MOVZWL  -142(CCB), R0                                        ; 0579
                                           15  13 00012            BEQL    1$                                                   ;
                                    FF7C   CB  DD 00014            PUSHL   -132(CCB)                                            ; 0582
                                           50  DD 00018            PUSHL   R0                                                   ;
                  00000000G  00            02  FB 0001A            CALLS   #2, FOR$$FREE_VM                                     ;
                                    FF72   CB  B4 00021            CLRW    -142(CCB)                                            ; 0583
                                    FF7C   CB  D4 00025            CLRL    -132(CCB)                                            ; 0584
              07        FF   AB            04  E1 00029 1$:        BBC     #4, -1(CCB), 2$                                      ; 0591
                                           6E  DD 0002E            PUSHL   LOGICAL_UNIT                                         ; 0593
                     0000V  CF             01  FB 00030            CALLS   #1, DEALLOCATE                                       ;
              10 00000000'  EF             6E  E4 00035 2$:        BBSC    LOGICAL_UNIT, FOR$$V_IOINPROG, 3$                    ; 0602
                 FFFFFFFB   8F             6E  D1 0003D            CMPL    LOGICAL_UNIT, #-5                                    ; 0604
                                           07  13 00044            BEQL    3$                                                   ;
                  00000000G  00            00  FB 00046            CALLS   #0, FOR$$SIG_DATCOR                                  ; 0606
```

```
                                          5B  D4 0004D 3$:        CLRL    CCB                                                ; 0608
                               5E         04  C0 0004F            ADDL2   #4, SP                                             ; 0612
                                              05 00052            RSB                                                        ;

; Routine Size:  83 bytes,    Routine Base:  _FOR$CODE + 0117


;  552          0613  1
```

```
ROUTINE DEALLOCATE (%SBTTL'Deallocate a CCB'
        LOGICAL_UNIT                            ! The LUN on which to deallocate
    ) : CALL_CCB NOVALUE =

!++
! FUNCTIONAL DESCRIPTION:
!
!       Release the heap storage associated with a CCB.  This is done after
!       a CLOSE.  If the file is an internal file, insert the LUB on
!       INTFIL_QUEUE rather than deallocating it.
!
! CALLING SEQUENCE:
!
!       DEALLOCATE (.LOGICAL_UNIT)
!
! FORMAL PARAMETERS:
!
!       LOGICAL_UNIT.rl.v       The LUN for which to deallocate the CCB
!
! IMPLICIT INPUTS:
!
!       INTFIL_QUEUE
!       Several fields of the LUB
!
! IMPLICIT OUTPUTS:
!
!       INTFIL_QUEUE
!       FOR$$A_LUB_TAB [.LOGICAL_UNIT] is cleared
!
! SIDE EFFECTS:
!
!       Deallocates heap storage
!
!--

    BEGIN

    BUILTIN
        INSQUE,
        TESTBITCC;

    EXTERNAL REGISTER
        CCB : REF $FOR$CCB_DECL;

    !+
    ! Split depending on whether or not this is an internal file/ENCODE/DECODE.
    !-

    IF .CCB [LUB$W_LUN] NEQ LUB$K_LUN_ENCD
    THEN
        BEGIN

        !+
        ! Remove this LUB from the LUB table.
        !-

        FOR$$AA_LUB_TAB [.LOGICAL_UNIT] = 0;
```

```
	!+
	! Deallocate record buffer, if present.
	!-

	IF (( NOT .CCB [LUB$V_USER_RBUF]) AND (.CCB [LUB$A_UBF] NEQA 0))
	THEN
	    FOR$$FREE_VM (.CCB [LUB$W_RBUF_SIZE], .CCB [LUB$A_UBF]);

	!+
	! Deallocate FAB if allocated by ASSIGN/FDBSET.  If filename
	! also allocated, deallocate it.
	!-

	IF .CCB [LUB$A_FAB] NEQA 0
	THEN
	    BEGIN
	    LOCAL
		HEAP_FAB: REF BLOCK [, BYTE];
	    HEAP_FAB = .CCB [LUB$A_FAB];
	    IF .HEAP_FAB [FAB$B_FNS] NEQU 0
	    THEN
		FOR$$FREE_VM (.HEAP_FAB [FAB$B_FNS], .HEAP_FAB [FAB$L_FNA]);
	    FOR$$FREE_VM (.HEAP_FAB [FAB$B_BLN], .HEAP_FAB);
	    END;

	!+
	! Deallocate resultant name string, if present.
	!-

	IF (.CCB [LUB$V_VIRT_RSN])
	THEN
	    FOR$$FREE_VM (.CCB [LUB$B_RSL], .CCB [LUB$A_RSN]);

	!+
	! Deallocate RFA cache, if present.
	!-

	IF .CCB [LUB$A_RFA_CACHE_BEG] NEQA 0
	THEN
	    FOR$$FREE_VM ((RCE_K_CACHE_SIZE * RCE_S_RCE_STRUCT),
		.CCB [LUB$A_RFA_CACHE_BEG]);

	!+
	! Deallocate LUB memory.
	!-

	FOR$$FREE_VM ((ISB$K_ISB_LEN + LUB$K_LUB_LEN + RAB$C_BLN +
	    FAB$C_BLN + NAM$C_BLN), .CCB - (ISB$K_ISB_LEN + LUB$K_LUB_LEN));

	RETURN;
	END;

    !+
    ! This is an ENCODE/DECODE/internal file. Insert the LUB on the queue.
    ! Use the first two longwords of the ISB as the queue link.
    !-
```

```
;  668          0728  2
;  669          0729  2      INSQUE (.CCB - (ISB$K_ISB_LEN + LUB$K_LUB_LEN), INTFIL_QUEUE);
;  670          0730  2
;  671          0731  2      RETURN;
;  672          0732  2
;  673          0733  1      END;



                                        000C 00000 DEALLOCATE:
                                                                .WORD    Save R2,R3                                       ; 0614
                     53 00000000G  00   9E 00002                MOVAB    FOR$$FREE_VM, R3
           FFFB      8F       C6   AB   B1 00009                CMPW     -58(CCB), #-5                                    ; 0662
                                   6B   13 0000F                BEQL     6$
                     50       04   AC   D0 00011                MOVL     LOGICAL_UNIT, R0                                 ; 0670
                        00000000'EF40   D4 00015                CLRL     FOR$$AA_LUB_TAB+32[R0]
                              FF   AB   95 0001C                TSTB     -1(CCB)                                          ; 0676
                                   0F   19 0001F                BLSS     1$
                              9C   AB   D5 00021                TSTL     -100(CCB)
                                   0A   13 00024                BEQL     1$
                              9C   AB   DD 00026                PUSHL    -100(CCB)                                        ; 0678
                     7E       D2   AB   3C 00029                MOVZWL   -46(CCB), -(SP)
                     63            02   FB 0002D                CALLS    #2, FOR$$FREE_VM
                              E8   AB   D5 00030 1$:            TSTL     -24(CCB)                                         ; 0685
                                   1C   13 00033                BEQL     3$
                     52       E8   AB   D0 00035                MOVL     -24(CCB), HEAP_FAB                               ; 0690
                              34   A2   95 00039                TSTB     52(HEAP_FAB)                                     ; 0691
                                   0A   13 0003C                BEQL     2$
                              2C   A2   DD 0003E                PUSHL    44(HEAP_FAB)                                     ; 0693
                     7E       34   A2   9A 00041                MOVZBL   52(HEAP_FAB), -(SP)
                     63            02   FB 00045                CALLS    #2, FOR$$FREE_VM
                                   52   DD 00048 2$:            PUSHL    HEAP_FAB                                         ; 0694
                     7E       01   A2   9A 0004A                MOVZBL   1(HEAP_FAB), -(SP)
                     63            02   FB 0004E                CALLS    #2, FOR$$FREE_VM
                     0A       FE   AB   E9 00051 3$:            BLBC     -2(CCB), 4$                                      ; 0701
                              F8   AB   DD 00055                PUSHL    -8(CCB)                                          ; 0703
                     7E       F7   AB   9A 00058                MOVZBL   -9(CCB), -(SP)
                     63            02   FB 0005C                CALLS    #2, FOR$$FREE_VM
                              C8   AB   D5 0005F 4$:            TSTL     -56(CCB)                                         ; 0709
                                   0B   13 00062                BEQL     5$
                              C8   AB   DD 00064                PUSHL    -56(CCB)                                         ; 0712
                     7E     0190   8F   3C 00067                MOVZWL   #400, -(SP)                                      ; 0711
                     63            02   FB 0006C                CALLS    #2, FOR$$FREE_VM
                            FEE0   CB   9F 0006F 5$:            PUSHAB   -288(CCB)                                        ; 0719
                     7E     0214   8F   3C 00073                MOVZWL   #532, -(SP)                                      ; 0718
                     63            02   FB 00078                CALLS    #2, FOR$$FREE_VM
                                   04   04 0007B                RET                                                       ; 0664
      00000000'      EF     FEE0   CB   0E 0007C 6$:            INSQUE   -288(CCB), INTFIL_QUEUE                          ; 0729
                                        04 00085                RET                                                       ; 0733

; Routine Size:  134 bytes,    Routine Base:  _FOR$CODE + 016A


;  674          0734  1
```

```
0735  1 GLOBAL ROUTINE FOR$$CB_GET %SBTTL'GET current CCB'
0736  1     : JSB_CB_GET NOVALUE =
0737  1
0738  1 !++
0739  1 ! FUNCTIONAL DESCRIPTION:
0740  1 !
0741  1 !       FOR$$CB_GET gets the curents LUB/ISB/RAB.
0742  1 !       This routine is only called from non-shared procedures which
0743  1 !       can't access FOR$$A_CUR_LUB directly.  (Entry vectors for
0744  1 !       data would mean that the code would have to change when the
0745  1 !       decision to make a module shared or non-shared is changed.
0746  1 !       Unless the LINKER got smarter and changed the level of indirection
0747  1 !       on data references which were vectored.)
0748  1 !
0749  1 ! CALLING SEQUENCE:
0750  1 !
0751  1 !       JSB FOR$$CB_GET ()
0752  1 !
0753  1 ! FORMAL PARAMETERS:
0754  1 !
0755  1 !       NONE
0756  1 !
0757  1 ! IMPLICIT INPUTS:
0758  1 !
0759  1 !       FOR$$A_CUR_LUB          Adr. of current LUB/ISB/RAB
0760  1 !
0761  1 ! IMPLICIT OUTPUTS:
0762  1 !
0763  1 !       CCB                     Set to adr. of current LUB/ISB/RAB.
0764  1 !
0765  1 ! RETURN VALUE:
0766  1 !
0767  1 !       NONE
0768  1 !
0769  1 ! SIDE EFFECTS:
0770  1 !
0771  1 !       NONE
0772  1 !--
0773  1
0774  2     BEGIN
0775  2
0776  2     EXTERNAL REGISTER
0777  2         CCB : REF $FOR$CCB_DECL;
0778  2
0779  2     CCB = .FOR$$A_CUR_LUB;
0780  2
0781  2     RETURN
0782  2
0783  1     END;                                    ! End of FOR$$CB_GET routine



                 5B 00000000'  EF  D0 00000 FOR$$CB_GET::
                                                MOVL    FOR$$A_CUR_LUB, CCB                                ; 0779
                                   05 00007     RSB                                                        ; 0783
```

```
; Routine Size:  8 bytes,    Routine Base:  _FOR$CODE + 01F0

;  725          0784  1
```

```
0785  1  GLOBAL ROUTINE FOR$$CB_FETCH (%SBTTL'Fetch a LUB, or 0'
0786  1        LUN                                          ! LUN of the LUB
0787  1     ) : CALL_CCB NOVALUE =
0788  1
0789  1  !++
0790  1  ! FUNCTIONAL DESCRIPTION:
0791  1  !
0792  1  !       FOR$$CB_FETCH returns the CCB address for a given LUN without
0793  1  !       "pushing" it.  This is used by FOR$$CLOSE_ALL and FOR$INQUIRE.
0794  1  !       ASTs must be disabled before FOR$$CB_FETCH is called and not
0795  1  !       reenabled until after the CCB is no longer needed.
0796  1  !
0797  1  ! CALLING SEQUENCE:
0798  1  !
0799  1  !       CALL FOR$$CB_FETCH (LUN)
0800  1  !
0801  1  ! FORMAL PARAMETERS:
0802  1  !
0803  1  !       LUN.rl.v                Logical Unit Number at which to "peek"
0804  1  !
0805  1  ! IMPLICIT INPUTS:
0806  1  !
0807  1  !       FOR$$V_LUN_OWNR         Table of LUN owners
0808  1  !       FOR$$AA_LUB_TAB         Table of pointers to LUBs
0809  1  !
0810  1  ! IMPLICIT OUTPUTS:
0811  1  !
0812  1  !       CCB                     This register is set to 0 if the LUN is not owned by FORTRAN
0813  1  !                               or is not allocated, or to the address of the LUB/ISB/RAB
0814  1  !                               otherwise.
0815  1  !
0816  1  ! RETURN VALUE:
0817  1  !
0818  1  !       NONE
0819  1  !
0820  1  ! SIDE EFFECTS:
0821  1  !
0822  1  !       NONE
0823  1  !--
0824  1
0825  2     BEGIN
0826  2
0827  2     EXTERNAL REGISTER
0828  2        CCB : REF $FOR$CCB_DECL;
0829  2
0830  2     CCB = .FOR$$AA_LUB_TAB [.LUN];
0831  2
0832  2     RETURN;
0833  1     END;                                          ! of routine FOR$$CB_FETCH



                              0000 00000        .ENTRY  FOR$$CB_FETCH, Save nothing                 : 0785
         50         04  AC  D0 00002        MOVL    LUN, R0                                     : 0830
         5B 00000000'EF40   D0 00006        MOVL    FOR$$AA_LUB_TAB+32[R0], CCB                 :
```

```
                                                            04 0000E          RET                                                ; 0833

; Routine Size:  15 bytes,    Routine Base:  _FOR$CODE + 01F8


;  776          0834  1
```

```
1 GLOBAL ROUTINE FOR$$NEXT_LUN (%SBTTL'Get next LUN which might be open'
1           FLAG: REF VECTOR [, LONG],              ! First-time and last-time flag
1           LUN: REF VECTOR [, LONG]                ! Logical Unit Number
1       ) : NOVALUE =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !         FOR$$NEXT_LUN gets a LUN which might be open.  It is used by
1 !         the exit handler declared by FORTRAN OPEN, which must look
1 !         through all the LUNs and do the DELETE or PRINT handling by
1 !         calling CLOSE.  (RMS close won't do DELETE or PRINT handling.)
1 !         This routine scans the table of LUB pointers and returns those
1 !         which are non-zero.  The caller must use CB_PUSH and CB_POP
1 !         to obtain control of the LUB.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !         CALL FOR$$NEXT_LUN (FLAG, LUN)
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !         FLAG.mv.r                If 0 on entry, this is the first call
1 !                                  and LUN is invalid.  If 1 on entry, LUN
1 !                                  is the last LUN processed.  On exit, 0
1 !                                  means that there are no more LUNs, and 1
1 !                                  means that LUN contains the Logical Unit
1 !                                  Number to process.
1 !         LUN.ml.r                 Logical Unit Number, as described above.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !         FOR$$AA_LUB_TAB
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !         NONE
1 !
1 ! RETURN VALUE:
1 !
1 !         NONE
1 !
1 ! SIDE EFFECTS:
1 !
1 !         NONE
1 !--
1
2     BEGIN
2
2     LOCAL
2         LOCAL_LUN;
2
2     !+
2     ! If this is the first entry, arrange to return the first logical
2     ! unit.
2     !-
2
```

```
;  835          0892  2      IF NOT .FLAG [0]
;  836          0893  2      THEN
;  837          0894  3          BEGIN
;  838          0895  3          FLAG [0] = 1;
;  839          0896  3          LOCAL_LUN = LUB$K_ILUN_MIN;
;  840          0897  3          END
;  841          0898  2      ELSE
;  842          0899  3          BEGIN
;  843          0900  3          LOCAL_LUN = .LUN [0] + 1;
;  844          0901  2          END;
;  845          0902  2
;  846          0903  2      !+
;  847          0904  2      ! While the unit number is in range, look for a LUB entry that is
;  848          0905  2      ! non-zero.
;  849          0906  2      !-
;  850          0907  2
;  851          0908  2      WHILE (.LOCAL_LUN LEQ LUB$K_LUN_MAX) DO
;  852          0909  3          BEGIN
;  853          0910  3          IF .FOR$$AA_LUB_TAB [.LOCAL_LUN] NEQ 0
;  854          0911  3          THEN
;  855          0912  4              BEGIN
;  856          0913  4              LUN [0] = .LOCAL_LUN;
;  857          0914  4              RETURN;
;  858          0915  3              END;
;  859          0916  3          LOCAL_LUN = .LOCAL_LUN + 1;
;  860          0917  2          END;
;  861          0918  2
;  862          0919  2      !+
;  863          0920  2      ! We dropped out of the loop.  Return failure.
;  864          0921  2      !-
;  865          0922  2
;  866          0923  2      FLAG [0] = 0;
;  867          0924  2
;  868          0925  2      RETURN;
;  869          0926  1      END;                                        ! End of FOR$$NEXT_LUN routine



                                         0000 00000           .ENTRY  FOR$$NEXT_LUN, Save nothing                      ; 0835
                   09         04     BC   E8 00002            BLBS    @FLAG, 1$                                        ; 0892
              04   BC                01   D0 00006            MOVL    #1, @FLAG                                        ; 0895
                   50                08   CE 0000A            MNEGL   #8, LOCAL_LUN                                    ; 0896
                                     05   11 0000D            BRB     2$                                               ; 0892
        50    08   BC                01   C1 0000F  1$:       ADDL3   #1, @LUN, LOCAL_LUN                              ; 0900
         00000077  8F                50   D1 00014  2$:       CMPL    LOCAL_LUN, #119                                  ; 0908
                                     12   14 0001B            BGTR    4$                                               ;
                        00000000'EF40     D5 0001D            TSTL    FOR$$AA_LUB_TAB+32[LOCAL_LUN]                    ; 0910
                                     05   13 00024            BEQL    3$                                               ;
              08   BC                50   D0 00026            MOVL    LOCAL_LUN, @LUN                                  ; 0913
                                          04 0002A            RET                                                      ; 0912
                                     50   D6 0002B  3$:       INCL    LOCAL_LUN                                        ; 0916
                                     E5   11 0002D            BRB     2$                                               ; 0908
                              04     BC   D4 0002F  4$:       CLRL    @FLAG                                            ; 0923
                                          04 00032            RET                                                      ; 0926
```

```
; Routine Size:  51 bytes,    Routine Base:  _FOR$CODE + 0207

;  870          0927  1
```

```
1 %SBTTL'FOR$$FP_MATCH - Find current incarnation'
1 GLOBAL ROUTINE FOR$$FP_MATCH (
1           SIG_FP                                  ! of ISB that has SIG_FP
1     ) : CALL_CCB NOVALUE =                        ! in ISB$A_USER_FP
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !         FOR$$FP_MATCH is part of the I/O in progress handling scheme.
1 !         It is called with one argument, the value of the frane pointer
1 !         desired.  It looks through the current ISB chain until it finds
1 !         an ISB that has the desired FP in ISB$A_USER_FP.  This means that
1 !         that ISB was the one in effect when the I/O in progress handler
1 !         was established.  If it finds one, external register CCB is set
1 !         to the CCB of that ISB.  If no match is found, there is something
1 !         seriously wrong in the database so error OTS$_INTDATCOR is
1 !         signalled.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !         CALL FOR$$FP_MATCH (SIG_FP)
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !         SIG_FP.rl.v              The FP present in the signal mechanism
1 !                                  list when the I/O in progress handler
1 !                                  was signalled.  This value is searched for
1 !                                  in the current ISB chain.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !         FOR$$AA_LUB_TAB          Table of pointers to LUBs.
1 !         FOR$$A_CUR_LUB           Address of current LUB.
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !         CCB                      This register is set to the address of the
1 !                                  ISB/LUB/RAB block that has SIG_FP in its
1 !                                  ISB$A_USER_FP.
1 !
1 ! RETURN VALUE:
1 !
1 !         NONE
1 !
1 ! SIDE EFFECTS:
1 !
1 !         Signals OTS$_INTDATCOR (Internal data corrupted in Run-Time Library)
1 !         if no ISB is found that matches SIG_FP.
1 !--
1
2     BEGIN
2
2     EXTERNAL REGISTER
2         CCB : REF $FOR$CCB_DECL;
2
2     LOCAL
2         LOGICAL_UNIT;                             ! Logical unit number of current LUB
```

```
;  929          0985  2
;  930          0986  2         !+
;  931          0987  2         ! Get current LUB
;  932          0988  2         !-
;  933          0989  2
;  934          0990  2         CCB = .FOR$$A_CUR_LUB;
;  935          0991  2
;  936          0992  2         !+
;  937          0993  2         ! Search through ISB chain to find matching FP
;  938          0994  2         !-
;  939          0995  2
;  940          0996  2         WHILE .CCB NEQ 0 DO
;  941          0997  3             BEGIN
;  942          0998  3             LOGICAL_UNIT = .CCB [LUB$W_LUN];
;  943          0999  3
;  944          1000  3             IF .CCB [ISB$A_USER_FP] EQL .SIG_FP
;  945          1001  3             THEN
;  946          1002  3                 RETURN;
;  947          1003  3
;  948          1004  3             CCB = .CCB [ISB$A_PREVIOUS_LUB];
;  949          1005  2             END;
;  950          1006  2
;  951          1007  2         !+
;  952          1008  2         ! If we get here, then there must not have been a match.
;  953          1009  2         ! This should never happen, therefore signal an error.
;  954          1010  2         !-
;  955          1011  2
;  956          1012  2         FOR$$SIG_DATCOR ();
;  957          1013  2         RETURN;
;  958          1014  1         END;



                                    0000 00000         .ENTRY  FOR$$FP_MATCH, Save nothing                    ; 0929
                  5B 00000000'  EF  D0 00002           MOVL    FOR$$A_CUR_LUB, CCB                            ; 0990
                                13  13 00009 1$:       BEQL    2$                                             ; 0996
                  50        C6  AB  32 0000B           CVTWL   -58(CCB), LOGICAL_UNIT                         ; 0998
            04    AC      FF4C  CB  D1 0000F           CMPL    -180(CCB), SIG_FP                              ; 1000
                                0E  13 00015           BEQL    3$                                             ;
                  5B      FF48  CB  D0 00017           MOVL    -184(CCB), CCB                                 ; 1004
                                EB  11 0001C           BRB     1$                                             ; 0996
     00000000G    00            00  FB 0001E 2$:       CALLS   #0, FOR$$SIG_DATCOR                            ; 1012
                                    04 00025 3$:       RET                                                    ; 1014

; Routine Size:  38 bytes,    Routine Base:  _FOR$CODE + 023A
```

```
1 %SBTTL 'INITIALIZE_INTFIL_QUEUE - Initialize INTFIL_QUEUE'
1 ROUTINE INITIALIZE_INTFIL_QUEUE
1     : NOVALUE =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       Initializes INTFIL_QUEUE to be an empty queue.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       INITIALIZE_INTFIL_QUEUE ()
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       INTFIL_QUEUE
1 !       V_INTFIL_QUEUE_INIT
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       INTFIL_QUEUE
1 !       V_INTFIL_QUEUE_INIT
1 !
1 ! COMPLETION STATUS:
1 !
1 !       NONE
1 !
1 ! SIDE EFFECTS:
1 !
1 !       Makes INTFIL_QUEUE an empty queue.
1 !
1 ! SIGNALLED ERRORS:
1 !
1 !       NONE
1 !--
1
2     BEGIN
2
2     LOCAL
2       AST_STATUS;                                     ! Previous AST enable status
2
2     BUILTIN
2       TESTBITCS;
2
2     !+
2     ! Disable ASTs.
2     !-
2
2     AST_STATUS = $SETAST (ENBFLG = 0);
2
2     !+
2     ! If V_INTFIL_QUEUE_INIT is still clear, initialize INTFIL_QUEUE to
2     ! be an empty queue.  Set V_INTFIL_QUEUE_INIT.
```

```
; 1017          1072  2         !-
; 1018          1073  2
; 1019          1074  2         IF TESTBITCS (V_INTFIL_QUEUE_INIT)
; 1020          1075  2         THEN
; 1021          1076  3             BEGIN
; 1022          1077  3             INTFIL_QUEUE [0] = INTFIL_QUEUE;        ! Set forward link
; 1023          1078  3             INTFIL_QUEUE [1] = .INTFIL_QUEUE [0];   ! Set backward link
; 1024          1079  2             END;
; 1025          1080  2
; 1026          1081  2         !+
; 1027          1082  2         ! Reenable ASTs if previously enabled.
; 1028          1083  2         !-
; 1029          1084  2
; 1030          1085  2         IF .AST_STATUS EQL SS$_WASSET
; 1031          1086  2         THEN
; 1032          1087  2             $SETAST (ENBFLG = 1);
; 1033          1088  2
; 1034          1089  2         RETURN;
; 1035          1090  2
; 1036          1091  1         END;                                        ! End of routine INITIALIZE_INTFILQUEUE


                                                                        .EXTRN  SYS$SETAST

                                                    000C 00000 INITIALIZE_INTFIL_QUEUE:
                                                                        .WORD   Save R2,R3                              ; 1016
                                53 00000000G    00  9E 00002            MOVAB   SYS$SETAST, R3                          ;
                                52 00000000'    EF  9E 00009            MOVAB   INTFIL_QUEUE, R2                        ;
                                                7E  D4 00010            CLRL    -(SP)                                   ; 1067
                                63              01  FB 00012            CALLS   #1, SYS$SETAST                          ;
                07          08  A2              00  E2 00015            BBSS    #0, V_INTFIL_QUEUE_INIT, 1$             ; 1074
                                62              62  9E 0001A            MOVAB   INTFIL_QUEUE, INTFIL_QUEUE              ; 1077
                            04  A2              62  D0 0001D            MOVL    INTFIL_QUEUE, INTFIL_QUEUE+4            ; 1078
                                09              50  D1 00021 1$:        CMPL    AST_STATUS, #9                          ; 1085
                                                05  12 00024            BNEQ    2$                                      ;
                                                01  DD 00026            PUSHL   #1                                      ; 1087
                                63              01  FB 00028            CALLS   #1, SYS$SETAST                          ;
                                                    04 0002B 2$:        RET                                             ; 1091

; Routine Size:  44 bytes,    Routine Base:  _FOR$CODE + 0260
; 1038          1092  1 END                                                     ! End of module FOR$$CB
; 1039          1093  1
; 1040          1094  0 ELUDOM




                                                                FOR$$CB_RET==           FOR$$CB_POP

;                                       PSECT SUMMARY
;
;       Name                        Bytes                           Attributes
;
;  _FOR$DATA                            544  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
;  _FOR$CODE                            652  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)



;                               Library Statistics
;
;                                               -------- Symbols --------       Pages       Processing
;       File                                    Total   Loaded   Percent        Mapped      Time
;
;  _$255$DUA28:[SYSLIB]STARLET.L32;1             9776       23         0         581         00:01.0
;  _$255$DUA28:[FORRTL.OBJ]FORLIB.L32;1           711      192        27          52         00:00.5
;  _$255$DUA28:[FORRTL.OBJ]RTLLIB.L32;1            36        0         0           8         00:00.1




;                                       COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACE/LIS=LIS$:FORCB/OBJ=OBJ$:FORCB MSRC$:FORCB/UPDATE=(ENH$:FORCB)

; Size:           652 code + 544 data bytes
; Run Time:          00:17.3
; Elapsed Time:      00:43.8
; Lines/CPU Min:      3794
; Lexemes/CPU-Min:  14184
; Memory Used:  117 pages
; Compilation Complete
```

COMR50WD
LIS

FORDATEDS
LIS

FORDECOMO
LIS

FORCB
LIS

COMSETST
LIS

FORASSOC
LIS

FORCLOSEF
LIS

FORDATE
LIS

FORCLOSE
LIS

FORDECOMP
LIS

FORDELETE
LIS

COMRAD50
LIS

COMUSEREX
LIS

FORBITOPS
LIS

FORDEFINE
LIS

FORBACKSP
LIS

FORDISPA
LIS

FORCUTRT
LIS